Automatic speech recognition using context-dependent syllables

Jan Hejtmánek and Tomáš Pavelka

***Abstract*— In this work, we deal with advanced context-dependent automatic speech recognition (ASR) of Czech spontaneous talk using hidden Markov models (HMM). Context-dependent units (e.g. triphones, diphones) in ASR systems provide significant improvement against simple non-context-dependent units. However, the usage of triphones brings some problems that we must solve. Mainly it is the total number of such units in the recognition process. To overcome problems with triphones we experiment with syllables. The main part of this article shows problems with the implementation of syllables into the LASER (ASR system developed at Department of Computer Science and Engineering, Faculty of Applied Sciences) and results of the recognition process.**

I. Introduction

THIS document describes the way how to effectively use syllables as a context-dependent phonetic unit in automatic speech recognition. As we have shown in previous works [2], [3] context-dependent units (e.g. triphones, diphones) in ASR systems provide significant improvement against simple non-context-dependent units. To overcome problems with triphones we experiment with syllables.

Syllables are context-dependent and their number is much lower than triphones. We believe that using syllables it will lead to improvement in recognition time and accuracy.

II. Syllables

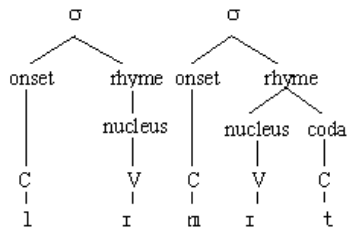

Fig. 1. Syllable in the root of the tree consists of optional onset and compulsory rhyme (rime). Rhyme than consist of compulsory nucleus and optional coda. C stands for "consonant" and V for "vowel"

From phonology definition, the syllable is a unit of pronunciation that consists of a central syllabic element (usually a vowel), that can be preceded and/or followed by none or more consonants [4]. The central syllabic element is called ***nucleus***, consonants that precede nucleus are ***onset,*** and consonants that follow nucleus are ***coda***.

The structure of syllables is a combination of allowable segments and typical sound sequences (which are language specific). These segments are shown in figure 1 with the example of English word "limit". The segments are made from consonants (C) and vowels (V). We distinguish four basic types of syllables.

A. Heavy syllables

Has a branching rhyme. All syllables with a branching nucleus (long vowels) are considered heavy. Some languages treat syllables with a short vowel (nucleus followed by a consonant (coda) as heavy.

B. Light syllables

Has a non-branching rhyme (short vowel). Some languages treat syllables with a short vowel (nucleus) followed by a consonant (coda) as light.

C. Closed syllables

Syllables end with a consonant coda.

D. Open

Has no final consonant.

Very short definition of syllables says that "syllables are the shortest pronounceable speech units" and the human creation and reception of speech is based mostly on syllables[1]. Moreover, suprasegmental[2] features of language affect the whole syllable and not any particular sound in the syllable.

The syllables are thus very eligible to be used as recognition units in ASR.

III. Syllabification

Syllabification is the separation of a word into syllables, whether spoken or written. It has very strict rules with many exceptions. The process of syllabification is however very complex and complicated.

We examined several basic algorithms for syllabification of written language. Because the LASER is ASR for Czech spoken language, we further worked only on the Czech syllabification process.

This work was supported by grant no. 2C06009 Cot-Sewing.

J. Hejtmánek, T. Pavelka, Laboratory of Intelligent Communication Systems, Dept. of Computer Science and Engineering, University of West Bohemia in Pilsen, Czech Republic
hejtman2@kiv.zcu.cz , tpavelka@kiv.zcu.cz

[1] Syllable-less languages do exist and even in every language there are exceptions ("shhh", "pssst", etc.). These exceptions however do not have direct strong impacts on findings in this work.

[2] Prosody, rythm, stress and intonation.

A. *Modified Liang algorithm*

The modified Liang algorithm is used in TeX word processors and is based on patterns. Patterns are made from words, syllables, and sets of characters by inserting scores between every character. After the dictionary of patterns has been made, the algorithm works in three easy steps:

1. Find all patterns that matches the input word
2. Insert the highest found score between every character
3. If the score between the characters is odd we can make syllable, if it is even we cannot.

We will take the Czech word "pejsek" (little dog) as an example in figure 2.

```
input word      p e j s e k
   ---------------------
patterns          2j1s2
                    1s e
                       e2k
   ---------------------
result          p e2j1s2e2k
```

Fig. 2. Syllabification of Czech word "pejsek" using Liang algorithm.

B. *Naive syllabification algorithm*

For comparison purposes, we build very simple rule-based syllabification algorithm.

This algorithm has only four steps:

1. Find all vowels, if two or more vowels are together group them.
2. Everything after last vowel (vowel group) belongs to the last syllable.
3. First character before every vowel (vowel group) belongs to this syllable.
4. Everything before the first vowel (vowel group) belongs to the first syllable.

C. *Lánský argorithm*

Thanks to [1] we managed to obtain working basis for English and Czech syllabification.

The process is very similar to our naive algorithm but it differs in the separation of consonants to the vowel groups:

1. Everything after the last vowel (vowel group) belongs to the last syllable
2. Everything before the first vowel (vowel group) belongs to the first syllable
3. If the number of consonants between vowels is even (2n), they are divided into the halves – first half belongs to the left vowel(s) and second to the right vowel(s) (n/n).
4. If the number of consonants between vowel(s) is odd (2n+1), we divide them into n/n+1 parts.
5. If there is only one consonant between vowels, it belongs to the left vowel(s).

We conducted two tests to find out how reliable the three methods are.

For the first test, very small text corpus was used. Three hundred words with the length of up to 18 characters ("Corpus 300") made from usual Czech words. The second test was conducted on our "train corpus". This is our testing corpus for ASR. It has 1460 distinct words, half of which are local names. Results of these two tests are shown in figure 3.

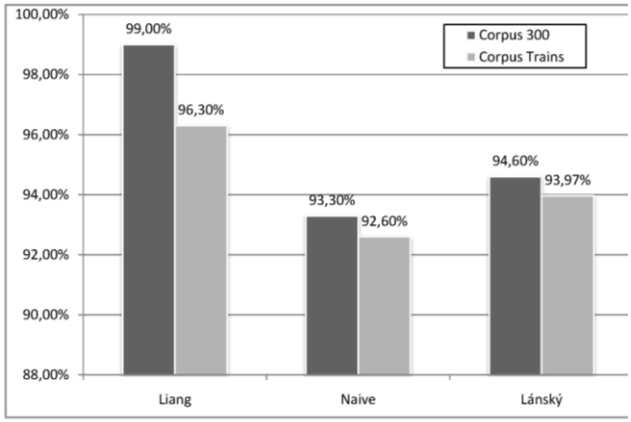

Fig. 3. Results of test of syllabification algorithms.

Reliability of an algorithm is computed as number of correctly syllabeled words divided by the total number of words. From the results, it is clear that the Liang algorithm is the best from our set of algorithms. The Lánský algorithm loses but only four percent.

IV. SYLLABIFICATION FOR ASR

All tests with syllabification have been conducted on orthographic transcriptions. However, the transcriptions used during the training and recognition process are phonetic transcriptions. For our purposes, the syllabification system had to be implemented on phonetic transcription.

Obtaining dictionary of right patterns for the Liang algorithm is highly problematic. Therefore, we decided to adapt the Lánský algorithm to work on either orthographic or phonetic transcriptions of Czech language. This adaptation gave us needed syllables and improvement in accuracy of syllabification. ***The accuracy rose*** from 93.97% to ***95.82%***.

Generally, the problems of the syllabification algorithm can be divided into two groups:

A. *The Root*

The root of the word is somehow exceptional and therefore the syllable was not recognized in 37 cases in the train corpus.

For example the word "Zábřeh" was divided into "Záb-řeh" instead of "Zá-břeh" where "břeh" (bank) is the root of the word.

B. *The Long numerals*

The Long numerals, which are composed of two or more basic numerals, are in Czech connected with "a" (like in "dvaadvacet" – twenty-two). These words should by split into syllables first around the connecting "a" and then like usual. For example the word "dvaadvacet" was split into

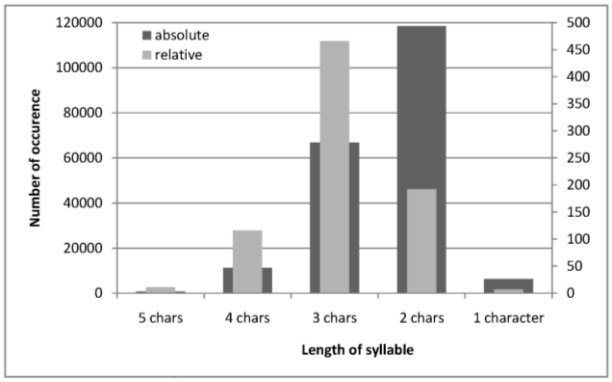

Fig. 4. Absolute and relative numbers of syllables in the train corpus.

"dvaad-vacet" instead of "dva-a-dva-cet". This systematic error led to 15 errors in the train corpus.

For the tests of ASR we didn't further improve the syllabification process. The statistics of syllables in the train corpus are shown in figure 4. These graphs document absolute and relative numbers of syllables in the corpus. The absolute number is the histogram of occurrences of syllables. In the relative number, every occurrence of every syllable is counted. It is visible that in relative numbers the three-character syllables are clearly the most common. But in the absolute numbers the most common are the two-character syllables. This was used during the test of recognizer.

V. Using Syllables in the ASR

Our LASER uses internal configuring file structure very similar to the one HTK (Hidden Markov Model Toolkit)[3] uses. Neither HTK nor LASER had the direct support for working with syllables we had to implement a transformation algorithm. This only transforms configuring files for monophone ASR into the form for syllable ASR.

The biggest problem of the triphones is the number of the units. To train such a huge number of units the training corpus has to be very large. The number of recognizable units in the train corpus can be seen in figure 5.

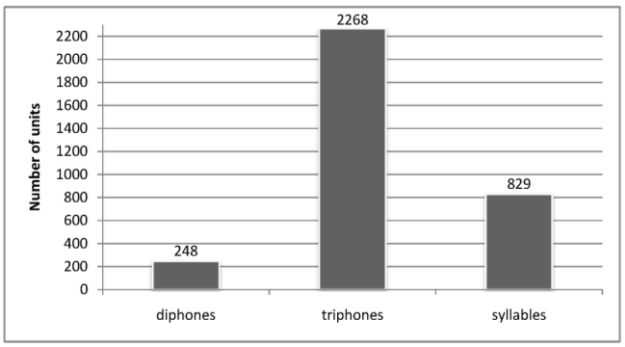

Fig. 5. Number of context-dependent units in the train corpus.

Using syllables instead of triphones we get both – relatively small number of recognizable units and context-dependency.

VI. Creating Models of Phonemes

To use the syllables in the HTK (LASER) recognizer it was necessary to adapt the models. First, the new models were built by concatenating monophone models to syllables. Thus models with variable number of states were created. These models will be referred to as "Syllables_var". For illustration see figure 6.

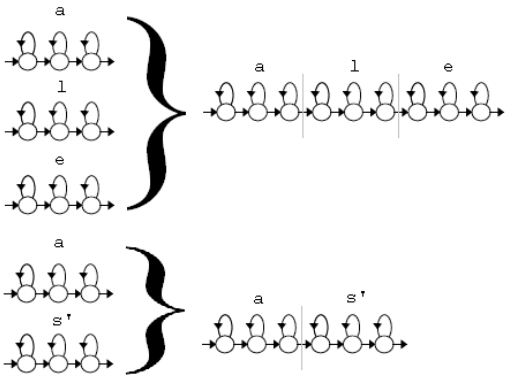

Fig. 6. Number of context-dependent units in the train corpus.

The monophone model is based on five-state HMM from which three states are emiting. Since the most common syllables in the train corpus are the two-character syllables, we build up the second testing model based on 7 state HMM (with five emiting states). These models will be refered to as "Syllables_5".

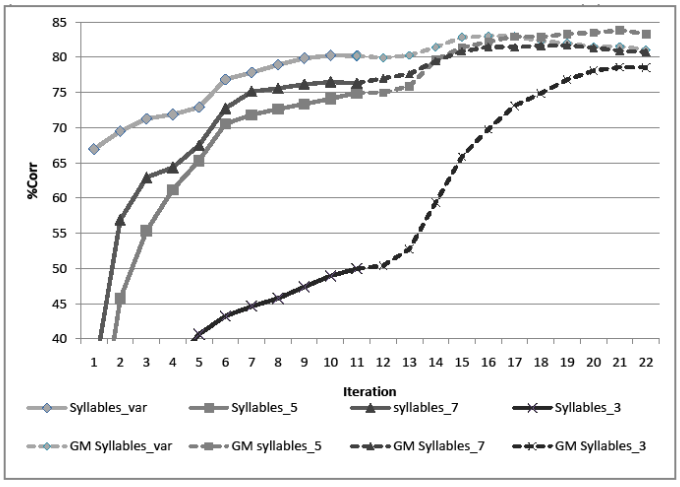

Fig. 7. Comparison of baseline tests.

[3] HTK is primarily used for speech recognition research. It has been used for numerous other applications including research into speech synthesis, character recognition and DNA sequencing.

VII. TESTS

A. Tests setup

The baseline test for both models is twelve iterations of training and testing. After this test we add Gaussian mixtures. Compared to [2] two mixtures are added in every iteration. The "Syllables_var" models were then tested with data-driven clustering in HTK with thresholds 50,100,150 and 250.

B. Comparison and measurements

We use three basic measures to compare results – Corr (Correct hits, in percents), Acc (Accuracy, in percents) and time (training and testing parts of every iteration, in percents).

All the tests were made on Intel C2D 6700 CPU, 4GB RAM, Windows XP Professional.

C. Tests results

Comparison of the baseline tests is shown on figure 7.

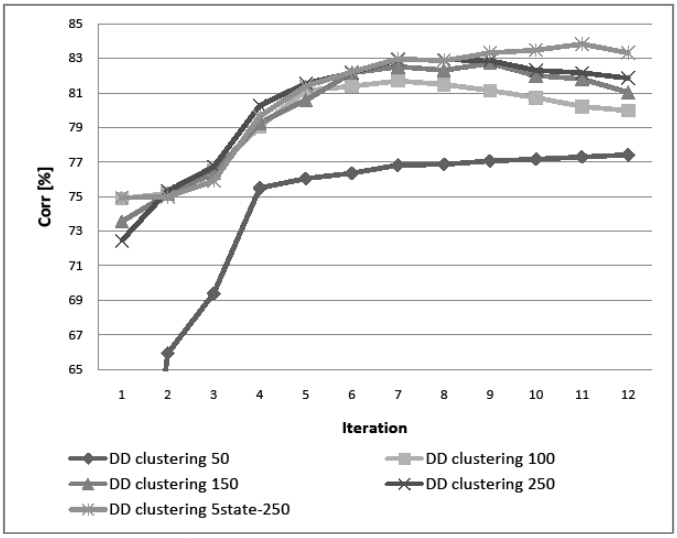

Fig. 8. Comparison of data-driven clustering tests.

Figure eight than adds time consumed during the training phase of single iteration. It is clearly visible that the less the unit has states the worse is the base test. However, the situation changes when we start adding Gaussian mixtures to the models. The addition of Gaussian mixtures helps best models with fewer states. This situation was expected. The higher is the number of emitting states in a model the heavier is the overprunning[4] of the language model. To avoid the overprunning we have to get more data for models. In this test it is achieved by data-driven and decision-tree clustering.

To confirm this theory several test were made with data-driven clustering. From our previous works we know that the data-driven clustering doesn't give as good results as decision-tree clustering but it is much easier to build the test. Results of this test are shown on figure 8. By lowering the number of real states to the 40% we get visible (yet little) progress against the baseline in Corr and time performance. The progress in correct hits is shown in figure 9.

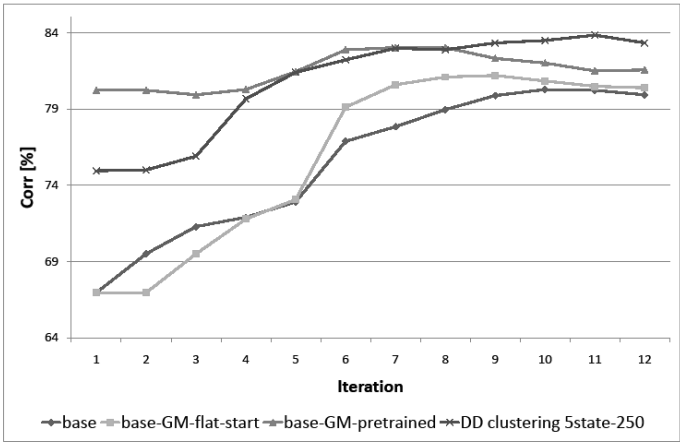

Fig. 9. Comparison of data-driven clustering tests.

Since all the tests were made in the very same conditions as the test made in [2] we can compare the results directly with the triphones results. These comparisons of baseline, Gaussian mixtures addition, and data-driven clustering are described in figure 11.

D. Decision-tree clustering

During the tests of decision-tree clustering, we run into several problems. The first problem is caused by the fact that when compared to triphones the syllable is one whole. This means that the questions in the decision tree are very hard to build. The second problem is to decide which part of the model is the right part to cluster – in decision-tree clustering, we ask for the context and from the answer we cluster the states. For example, in the "syllables_5" we have model "vlak". This model is built from monophones "v-l-a-k". We cannot tell which of the five emitting states the old monophone "a" is and we cannot clearly cluster the states.

We tried to solve both problems with model's set "Syllables_var". This model performs well (as seen in figure 7). Since this model has the highest number of states, it suffers heavily by over-training. According to our theory well built decision-tree clustering should make this particular model perform better then the triphone model set.

However, we were not able to build the decision tree yet. It is time-consuming work and to this date it is still unsolved. The basic decision-tree we have built proved that the it is possible to use the "Syllables_var" for clustering but the results were lower than anything we have presented.

[4] If the ASR has very little training data for a model it is stated as overprunning. The model is not trained further and the overall score is falling.

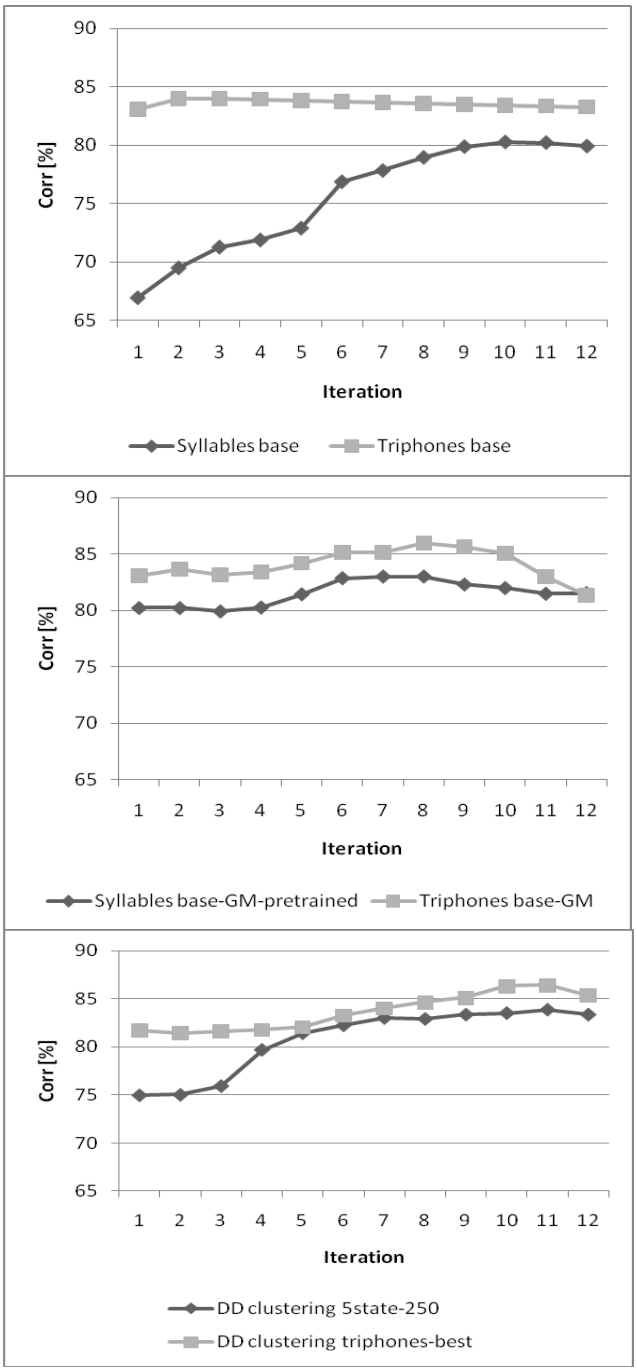

Fig. 10. Comparison of syllables-based versus triphons-based ASR.

VIII. CONCLUSION

We have successfully built several syllable-based ASR systems. Thanks to contex-dependency the baseline results were much higher than monophone ASR and slightly worse than fine-tuned triphone ASR. We have also successfully tested data-driven clustering, which led to visible improvement. From the part 3 (clustering part) of figure 10 it is visible that models in the iteration 8 and above are over-trained and better clustering is the key to get better performance. Future work will be building of decision-tree-based clustering. Preliminary results from this and previous works show as that it will lead to better performance of the ASR system.

IX. *REFERENCES*

[1] J. Lánský, M. Žemlička, "Text Compression: Syllables". *Proceedings of the Dateso 2005 Annual International Workshop on DAtabases, TExts, Specifications and Objects. CEUR-WS*, Vol. 129, pg. 32-45, ISBN 80-01-03204-3, ISSN 1613-0073

[2] J. Hejtmánek, Use of context-dependent units in speech recognition, Master thesis, University of West Bohemia in Pilsen, Faculty of Applied Sciences, 2007.

[3] Hejtmánek, J., Pavelka, T., *Use of context-dependent units in Czech speech*, Proc. of PhD Workshop 2007, Balatonfüred, Hungary, 2007

[4] S. Young, G. Evermann, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P.Woodland, *The HTK Book (for HTK Version 3.3)*, Cambridge University Engineering Department, 2005..

[5] K. Yu; J. Mason, J. Oglesby, "Speaker recognition models", *Proceedings of Eurospeech 95*, 1995, pp. 629-632

[6] M. Edgington et al., "Prosody and speech generation", *BT Technology Journal*, Volume 14 Number 1, 1996, pp. 84-99

[7] SIL International, "Glosary of linguistic Terms", www.sil.org, 2008